GUIDE

Software FMEA vs Hardware FMEA: What Actually Changes

> Hardware FMEA starts from physical components that degrade and fail at measurable rates. Software FMEA starts from functions, requirements and states: software does not wear out, so its failure modes are latent defects — missing requirements, unhandled conditions, bad state transitions and interface mismatches — triggered by specific inputs.

The fundamental difference

A resistor has a failure rate. A function does not. Every software failure was designed in, which means a software FMEA is really a systematic search for the conditions the design never accounted for.

	Hardware FMEA	Software FMEA
Item analyzed	Component or part	Function, requirement, interface or state
Failure cause	Wear, stress, manufacturing defect	Latent design or requirements defect
Failure rate	From parts databases and stress models	From defect prediction and historical defect data
Occurrence ranking	Physical probability	Likelihood the triggering condition occurs and escapes test
Detection	Inspection, BIT, monitoring	Reviews, static analysis, test coverage, runtime checks
Mitigation	Derating, redundancy, part change	New requirement, defensive design, exception handling, added test

Redundancy does not work the same way

Two identical hardware units rarely fail at the same instant. Two identical software copies fail identically, every time, on the same input. Software redundancy only helps when the redundant path is genuinely different — different algorithm, different data source, or a hardware-based safe state.

Typical software failure modes

- Missing or ambiguous requirement — behavior undefined for a real condition
- Unhandled input: out of range, null, malformed, out of order, duplicated
- Invalid state transition or startup/shutdown/restart in an unexpected state
- Timing: race condition, timeout too short or too long, missed deadline

- Resource exhaustion: memory, storage, handles, queue depth
- Interface mismatch: units, precision, endianness, protocol version
- Degraded-mode behavior after a dependency fails
- Error handling that fails, masks the fault, or loses data

A workable software FMEA process

1. Choose the level of analysis: functional, requirements-level, interface-level or detailed design.
2. List the items at that level and the intended behavior of each.
3. For each item, apply a failure mode checklist rather than free-form brainstorming — coverage comes from the checklist.
4. Record the local effect, system effect and end effect, including the mission or safety consequence.
5. Rank severity, occurrence and detection, or use a criticality scheme your program already applies.
6. Assign mitigations as requirements, design changes or specific tests — each with an owner.
7. Verify closure: every high-criticality mode maps to a test case or an accepted risk.

Common mistakes

- Analyzing code line by line, which does not scale and misses missing requirements entirely.
- Assuming a failure mode is impossible because the requirement says so — the requirement is what is being questioned.
- Producing a spreadsheet nobody converts into requirements or test cases.
- Skipping degraded modes, restart behavior and interfaces with other systems.

Frequently asked questions

▼ How is software FMEA different from hardware FMEA?

Software has no wear-out mechanism and no per-part failure rate. Software failure modes come from functions, requirements, states and interfaces, and every one traces to a latent design or requirements defect rather than physical degradation.

▼ Is there a software FMEA standard?

Yes. SAE J1025 Section 7 addresses software FMEA, and IEC 61508, ISO 26262 Part 6, SAE ARP5580, NASA-STD-8719.13 and NATO AOP-52 all address software failure modes analysis in their domains.

▼ What level should a software FMEA be performed at?

Functional or requirements level for most programs. Detailed design level is reserved for safety-critical components where the consequence of failure justifies the effort.

Related pages

Software FMEA training Software FMEA task SAE J1025 FMEA Section 7 ISO 26262 Part 6